

Quality assurance of CASE tools: a monitoring perspective

J.F. Coxhead & E.A. Dodman

Software Technology Group, Leeds Metropolitan University, Beckett Park Campus, Leeds, LS6 3QS, UK

ABSTRACT

Current methods of assuring the quality of software products concentrate on measurements of the quality of the end product and the management of the process by which the product is produced. However the achievement of quality is a complex matter dependent on many issues, for example the standard of the staff involved and the quality of the tools they use. Thus staff insufficiently trained will be unable to produce quality products even if good development tools are available. Conversely good staff will find it difficult to produce quality products if they do not have appropriate tools available. This is as relevant to an academic institution teaching and training software developers as it is for a commercial enterprise striving for Total Quality Management (TQM)

At LMU we have installed a configurable Software Engineering Environment, based on meta-case technology, to use in the education and training of software engineers. The educational philosophy is that students are taught design principles and methods (e.g. SSADM) and put these into practice using the environment.

The system has been used successfully with undergraduate and postgraduate students as evidenced by student enthusiasm and improved coursework. We now intend to quantitatively monitor and evaluate the environment's fitness for purpose and to build up profiles of normal student usage and progress. In the short term these profiles will be used for the early detection of students who are experiencing difficulties. In the long term they may be used to provide automatic triggers to a number of aids such as feedback, extra advice and supportive tuition or on-line corrective training.

650 Software Quality Management

Mechanisms have been devised to obtain quality assurance baselines covering tools in the system taking into account the differing stages of training of the students using them. These baselines have been used to devise the required quality profiles for the students and measures of how effective the environment is in enabling users to reach the required competencies. The paper discusses the criteria used and the implementation of the monitoring and assessment phase of the projects.

1.MONITORING AS A FACTOR IN IMPROVING QUALITY.

1.1. Total Quality Management (TQM)

The concept of Total Quality Management is a useful one but the use of the term TQM may conceal that it's achievement is a difficult and complex task. Current methods of measuring quality tend to concentrate on the end product or, as in the ISO 9000 series or TickIT, the process of production of the software. It is often mooted that commercial companies use such accreditation schemes largely as a marketing advantage. We have lengthy discussions and conferences about quality which promotes the idea that it is an issue that can be discussed in isolation whereas in truth it should be ingrained in everything we do.

Whatever our perception of quality might be it is safe to say that it will involve people and as such will be subject to human failings. In other words you may have the right tools and the right process to produce quality only to be let down (lose control) because of unpredictable human factors. The only way to understand the effects of human unpredictability is to monitor the software production process.

This is often done by using a set of matrix measures which again may only be obtained by having this ability to monitor. However these measures rarely if ever include criteria concerning the users in the process such as how good is that particular individual at doing the required task? or what is the minimum standard we can expect from a number of people doing the same job?

1.2 Including Human Criteria

The inclusion of such criteria is a vital ingredient towards TQM as it would allow for proper skills audits and consequently its analysis would be the basis of a realistic training program. If commercial companies could realise the benefits of such training programs being updated or modified as a result of their day to day monitoring of quality then there may be a drive to force tool vendors to start building more realistic teaching/training facilities into their tools, toolsets and environments.

1.3 Opposition To Human Criteria

Probably the biggest opposition to human criteria being included in any quality management system is fear by the subjects that any monitored results will be misused by the managers (The big brother syndrome). To get a workforce to change to accept this type of monitoring would mean managers having to win the trust of the individual with assurances that jobs would not be lost as a result of such monitoring.

1.4 Failure To Fulfil Tasks

It could be argued that an employee would not fulfil their allotted tasks for three main reasons and could all be put down to the responsibility of the company itself. If an employee cannot fulfil the task they are given then they may not understand what is required or they may not have the ability to do the job. In case of the first instance it could be put down to lack of adequate training in the second instance it could be put down to an inadequate recruiting policy. The third instance where an employee might fail is by lack of concentration or de-motivation for the task (brought on by any number of problems such as stress from over work, personality clash or personal problems). This being the case then line managers should accept some responsibility for welfare awareness or lack of it.

1.5 Monitoring In Education

In some respects the educational area is an ideal one in which to introduce monitoring. Students expect to be controlled and assessed and as they are in a learning situation an inability to perform particular tasks is an indication of current lack of knowledge and experience rather than a threat to their self-esteem. However there are particular problems in that students are asked to perform a wide variety of tasks whereas in industry the job description of a software engineer tends to be more circumscribed.

2. THE WORK AT LEEDS METROPOLITAN UNIVERSITY

2.1 Introduction

The work at Leeds Metropolitan University in regard to this paper is to explore the monitoring of undergraduate software engineers using a number of CASE tools. To get to this phase of the research a teaching environment had to be put in place [1] and this had to be supported by a teaching philosophy [2].

The importance of this work in support of the commercial enterprise is that graduates will be exposed to the idea of being monitored at an early stage in their career and should be more ready to expect it in their place of work. The other important aspect of this work is to ascertain the type of training facilities that would need to be available in CASE tools used in a commercial environment that was committed to TQM.

2.2 The Technology

In theory any tools used in the teaching of software engineers in academic institutions should not differ in functionality from their commercial counterparts. However, how these tools are used within the different enterprises are often very different. To resolve the problem of adapting tools built primarily for the commercial enterprise for acceptable usage within an academic enterprise required a special type of technology.

The technology used is meta-CASE and in this case a set of products produced by IPSYS Software plc. Meta-CASE tools are used to generate CASE tools. The intent of meta-CASE technology is to capture the specification of a required CASE tool and then generate the tool from the specification.

A significant breakthrough with meta-CASE is genericity and re-use at a high level. A good example of this is a graphic design editor which is usually required with most case tools. Usually each instance of such an editor is built from a lower level using components of some language library. With Meta-CASE the whole diagram editor can be built and re-used by parameterising it to support a number of different methods. This type of higher level re-use results in such things as generic interfaces across all tools and facilitates for standard messaging and help systems which will be already inherent in each instance of the editor you wish to parameterise. A more explicit description of Meta-CASE can be found in two papers by Alderson [4] [5].

2.3 Monitoring

The IPSYS tool building environment has a standard messaging system built into all its tool instances. If you build an IPSYS tool you automatically inherit a built-in messaging system with the ability to direct the messages to three different locations. The three locations are, the screen (stdout), a message area defined within the tool itself (display) and a session log. It is the exploitation of the session log option which gives us the monitoring capability. Within the IPSYS environment there are eleven different types of message sets (see fig 1) and each message within the set can be indexed.

fig 1 Ipsys Environment Message Sets

MESSAGE TYPE
1. Information
2. Warning
3. Error
4. Fatal
5. Comment
6. Metrics
7. Diagnostics
8. Tracking
9. Consoles
10. Security
11. System

2.4 Session Logs

When an IPSYS built tool is invoked it automatically kicks off a session log which records several mandatory messages such as tool start time and finish time along with any other messages that have been directed to the log from the 11 message set types (some of which are error messages). As a policy of our research we trap every possible piece of information. The idea being that if the information is not required currently we may need it at a latter date. It is easier to modify programs to query data that is already captured than it is to interrupt the system operation to capture new pieces of data. The session logs produce coded output in ascii format (see fig2)

fig 2 Session Log Output

```
13/04/1994:11:40:24:6789:standard:102:11:1:IPSYS SSADM 1.2 starts.
13/04/1994:11:40:40:6789:design_editor:240:1:2:Initialising internal tables.
13/04/1994:11:41:12:6789:design_editor:239:1:3:Accessing the database.
13/04/1994:11:42:23:6789:dbi:15:3:4:No such attribute : /study/isb2/isb219/assign1
13/04/1994:11:45:44:6789:se:23:1:5:No options available for selected area.
13/04/1994:11:46:34:6789:se:23:1:6:No options available for selected area.
13/04/1994:11:46:37:6789:se:23:1:7:No options available for selected area.
13/04/1994:11:52:57:6789:design_editor:223:2:8:There is no current operation to repeat.
13/04/1994:11:52:59:6789:design_editor:223:2:9:There is no current operation to repeat.
13/04/1994:12:01:32:6789:design_editor:219:3:10:The link must terminate at a node - try again
13/04/1994:12:01:33:6789:design_editor:219:3:11:The link must terminate at a node - try again
13/04/1994:12:01:34:6789:design_editor:219:3:12:The link must terminate at a node - try again
13/04/1994:12:01:40:6789:design_editor:219:3:13:The link must terminate at a node - try again
13/04/1994:12:01:45:6789:design_editor:220:3:14:The required link is not allowed - try again.
13/04/1994:12:02:05:6789:design_editor:223:2:15:There is no current operation to repeat.
13/04/1994:12:02:07:6789:design_editor:223:2:16:There is no current operation to repeat.
13/04/1994:12:02:42:6789:design_editor:223:2:17:There is no current operation to repeat.
13/04/1994:12:02:44:6789:design_editor:223:2:18:There is no current operation to repeat.
13/04/1994:12:08:49:6789:design_editor:219:3:19:The link must terminate at a node - try again
13/04/1994:12:08:50:6789:design_editor:219:3:20:The link must terminate at a node - try again
13/04/1994:12:16:10:6789:se:23:1:21:No options available for selected area.
13/04/1994:12:16:34:6789:se:23:1:22:No options available for selected area.
13/04/1994:12:16:35:6789:se:23:1:23:No options available for selected area.
13/04/1994:12:17:24:6789:standard:103:11:24:IPSYS SSADM 1.2 ends.
```

2.5 Session Log Control

An independent session log is started every time a student logs onto the system. This can amount to a significant number of session logs being created in a single day and consequently this uses a large amount of disk space and this would be unacceptable if allowed to build up over a period of time. The first task was to provide a means of concatenating all the session logs into one compressed datafile in a format suitable for later analysis.

The way in which this is managed is that the administrators (The course tutors whose classes are using the tools) of the system have a common (UNIX group) account. All the session logs that are created by the students are created in a common directory in the administrators account. The students have no access to the administrators account so they cannot tamper with the logs once created.

The owner of the account is the only person who can delete session logs which currently takes place once a day after conversion into their analytical format. This will eventually be an automatic process with each conversion being appended to the analysis data file. All other members of the group have read/write access to the session log directory.

2.6 Session Log Analysis

The goals for this part of the research are two fold. In the first instance the goal was to try and get some quality measure of the environment and the CASE tools/toolsets in their effectiveness as an integrated teaching environment.

From the analysis of the session logs profiles can be built up of student usage. The drop off of different types of error rates can indicate how easy or difficult the tools are to learn. All system faults are detectable so robustness can also be monitored at different stages of system loading.

This is all backed up by a questionnaire to determine the background of the students such as had they had any previous exposure to case tools, window management systems or other features of the environment. If this was the case it would have some obvious effect on any profiles that were being put together for future monitoring.

656 Software Quality Management

To qualify this analysis the session log datafile can be reformatted for input into a number of packages such as a statistics package, spreadsheet or even a database for more directed queries. What follows is three examples of such output along with some interpretation.

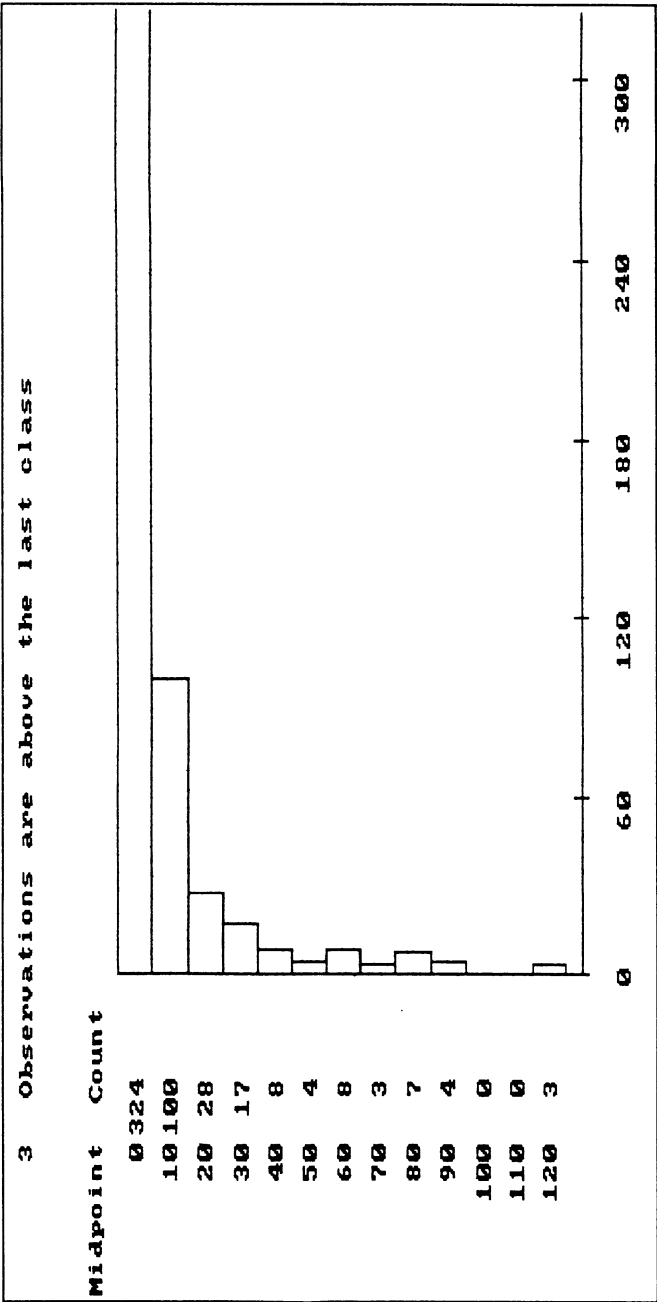
Example 1.

Figure 3 shows output from a statistics package (minitab) for a class of 16 students using an SSADM CASE tool for the first time and runs for a period of 2 hours.

The graph shows the number of messages invoked within each ten minute time slot relative to the tool start (i.e. 324 messages in the first ten minute and 100 messages between 10 and 20 minutes etc). It must be noted that all the messages are not errors (ref fig1).

The implication of the results would be dependent on the goal of the question to be answered. To give a flavour of this type of analysis, the drop off of messages after thirty minutes of tool use could imply that the tool was easy to learn. The profile of the graph could be as expected with messages reducing to none existent over time as students become familiar with the tool and then increasing again when they log off or become tired.

fig 3 Number of Messages Logged Over A Whole Session (Class Size 16) in Ten Minute Time Slots

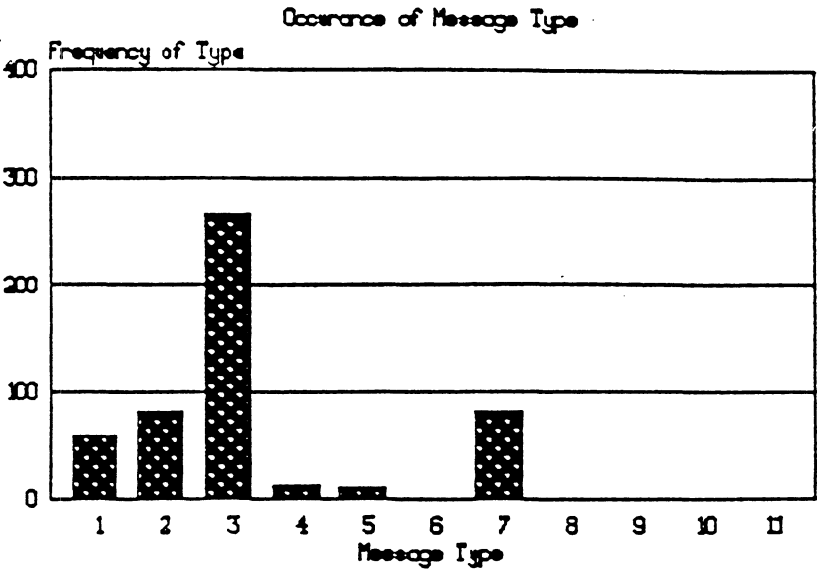




Example 2

This is again the same situation as previous but this time the output was passed into a spreadsheet and the total messages invoked in the two hours are sorted into types (fig4). By referencing figure number four to figure number one it can be seen that the messages were in fact a combination of information, warnings and errors. This was to be expected for first time student use of the tool. If a lot of system and fatal errors had shown up it would indicate that the tool was not robust enough to be used for teaching.

fig 4 Occurrence of Message Type (Spreadsheet)





Example 3

This example is a database perspective where a query is entered on-line. In figure 5 the tutor enters a student name and a date in which he/she is interested and in this case the system responds with the start and stop times of the tools which the student used within that day and a breakdown of the message types that were invoked.

fig 5 Database Perspective

```
User       : dimit
Date       : 22/02/1993
Process id : 289

Tool       : IPSYS SSADM 1.2.
Time Started : 15:37:27
Time Stopped : 16:03:53
```

Messages

```
-----
Informations : 0
Warnings    : 2
Errors      : 1
Fatal errors : 0
Comments    : 0
Metrics     : 0
Diagnostics : 0
Trackings   : 0
Console' s  : 0
Security' s : 0
System' s   : 0
```

...Strike enter to continue...

```
User       : dimit
Date       : 22/02/1993
Process id : 335

Tool       : IPSYS Information Engineering
Time Started : 16:04:22
Time Stopped : 16:22:40
```

Messages

```
-----
Informations : 0
Warnings    : 0
Errors      : 1
Fatal errors : 0
Comments    : 0
Metrics     : 0
Diagnostics : 0
Trackings   : 0
Console' s  : 0
Security' s : 0
System' s   : 0
```

...Strike enter to continue...

660 Software Quality Management

The notion long term is to capture a number of stable profiles of systems usage and then train a neural network to monitor it in real time triggering responses when any of the profiles exceed their normal limits.

2.7 Student Monitoring

The second goal of the research and probably the most important from our prospective is student monitoring. The long-term aim in this area is to provide a good basis for computer aided learning. It is envisaged that with the use of multi-media students will be able to learn and practice their CASE experience utilising computer aided learning. It is therefore essential that a good monitoring system is in place in order to

- * Allow for real-time monitoring of students to identify the strugglers at the earliest opportunity and either have tutor intervention or trigger some context related advice.

- * Provide tutor support with such things as assessment, appraisal, attendance and plagiarism. The use of a database with more directed queries as demonstrated in figure 5 would be a big advantage in this instance.

The error types relating to student usage that are trapped by the session logs are both tool usage and method errors so some estimations can be made on a students knowledge of the method they are using. The implications of this could be used to improve teaching effectiveness in this area.

3. Further Work

The monitoring aspects of the research as described above are now well advanced and can be demonstrated. The important piece of work that is still to be fully researched is how to monitor the design process. We can effectively monitor tool knowledge and method syntax but what is lacking is the ability to monitor the semantics of the design process. This is important if we are to get a more realistic picture of teaching effectiveness and student support.

A parallel piece of work that is taking place is the development of a graphical front-end for the monitoring system. This will be in the form of a Tutor management system where Tutors will be able to plan and execute their monitoring to requirement.

4. CONCLUSION

If commercial practice is to include TQM then it is doubtful whether it will be achievable without some monitoring taking place. If monitoring is inevitable then what has been discussed in this paper outlines and demonstrates how basic monitoring can be used to good effect.

A simple idea of being able to direct all tool messages out to a session log has been an added bonus to the already open flexibility of the IPSYS Meta-CASE technology being used at LMU. How nice it would be if tool vendors started putting such utilities in their tools as standard with an informed presumption that as well as their tools being used in their primary professional roles they would also be part of a TQM system.

REFERENCES:-

- [1] Coxhead.J.F., Harvey.J., Dodman.E.A.,(1991), "CASE Requirements for Teaching Software Engineering", Proc SEHE Conference, Southampton Institute.
- [2] Coxhead.J.F., Harvey.J., Dodman.E.A.,(1992), "An Integrated Environment and Teaching Philosophy for the Education of Software Engineers" Proc ISTIP Conference (1992).
- [3] Coxhead.J.F., Dodman.E.A., Harvey.J., 'CASE Technology: Improving Quality in the Education of Software Engineers', Proc Software Quality Management Conference (1993), Computational Mechanics Publications, Elsevier Science Publishers.
- [4] Alderson. A., 'Meta_CASE Technology' IPSYS Software plc, (1992).
- [5] Alderson. A., 'Beyond Today's CASE Technology Toward Meta_CASE', IPSYS Software plc, (1993).